2018 年 1 月 22 日作成

名古屋工業大学 先進セラミックス研究センター　井田 隆

2018 年 1 月 22 日更新

第３部

最適化とモンテカルロ法

８．最適化　Optimization

最適化（最適化計算） optimization とは，（コンピュータを使って）関数の最大あるいは最小を求めることです。技術や産業・経済だけでなく，政治の意味でも，教育の意味でも「最適化」は重要です。今までに学んだこと（第２部　構造のシミュレーション，ポテンシャルの計算など）と「最適化」を組み合わせることで，例えば「安定な構造の推定」を実現することができます。

高校数学の典型的な問題として，「微分」を使って極大や極小，最大や最小を求めるものがあります。積分は「高校の数学」では解けない場合があるのに対して，微分は「高校の数学」でも必ず解けます。ただし「関数の微分がゼロになる x を求める方程式」は「高校の数学」では解けないことがあります。そのような場合でもコンピュータを使って数値計算をすれば事実上解くことができます。

ポテンシャルを最小にする構造は「エネルギー的に最も安定な構造」と呼ばれることがあります。

一定の圧力で最も安定な構造は，エンタルピー enthalpy

$$H = U + PV \tag{8.3}$$

を最小にする構造として求められます。ここで U は内部エネルギー，P は圧力，V は体積です。周期的境界条件を用いる場合には，U として「基本セル当たりの内部エネルギー」，V として「基本セルの体積」を用い，「基本セルあたりのエンタルピー」について議論することになります。

一定温度，一定圧力で最も安定な構造はギブスの自由エネルギー

$$G = H - TS = U + PV - TS \tag{8.4}$$

を最小にする構造になるはずです。ここで T は温度， S はエントロピーです。ただし，ダイナミックスのシミュレーションのためには運動方程式（あるいは発展方程式）を解く方法が使われます。

最適化は，コンピュータを使った構造シミュレーションだけではなく，実験データ（力学的／光学的／電気的／磁気的な測定の結果）を解析する場合にも用いられます。

実験データを解析する場合の基本的な考え方として，ベイズ推定，最大事後確率推定，最尤推定（さいゆうすいてい），最小二乗法などがあります。最尤推定とは，「観測されたデータの出現する確率が最大となるようなモデルを求める」ことを意味します。構造モデルなどの理論モデルによって，実験データの出現する確率を関数の形で表すことができれば，「実験データの解釈」は，この関数の最大を求める問題と同じことです。

つまり，

安定構造のシミュレーション　＝　ポテンシャル関数の最小化

最尤推定による実験データ解析　＝　尤度関数（データの出現確率）の最大化

という関係があります。

任意の n 変数関数 $f\left(x_1, \cdots, x_n\right)$ の最大化は，関数 $g\left(x_1, \cdots, x_n\right) = -f\left(x_1, \cdots, x_n\right)$ の最小化と同じことです。以下任意の関数の最小を求める問題（方法）に限定して考えます。

8－1　最適化の方法　Methods for optimization

コンピュータを使った最適化の代表的な方法に以下のようなものがあります。

（1）グリッド・サーチ法

（2）黄金分割法（一次元の最小化）とそれを使った多次元の最小化の方法

（3）滑降型シンプレックス法

（4）モンテカルロ法と関連する方法（乱数を使う方法）

（5）ニュートン法と関連する方法

グリッド・サーチ grid-search 法とは，関数の変数が取りうる範囲を細かく区切り，すべての組み合わせで関数を評価し，最小の値をとる組み合わせを選ぶ方法です。効率が悪く，得られる結果の精度も高くありませんが，最も安全で確実な方法とも言えます。他の方法で最小を求めるとしても，「正しく最小が得られているかを確認する」などの目的で有効なので，知っておくべきでしょう。

黄金分割法と滑降型シンプレックス法については次節以降で，モンテカルロ法と関連する方法については次章で扱うことにします。ニュートン法は基本的に「微分がゼロになるような解」を求める方法で，多くの他の教科書で取り上げられていますが，ここでは省略します。

8－2　黄金分割法　Golden section method

$$r = \frac{2}{1+\sqrt{5}} = \frac{\sqrt{5}-1}{2} = 0.61803\cdots \tag{8.2.1}$$

あるいは,

$$r' = \frac{1}{r} = \frac{\sqrt{5}+1}{2} = 1.61803\cdots \tag{8.2.2}$$

を黄金比 golden ratio と呼びます。また，短辺と長辺の比が黄金比となる長方形を黄金長方形と呼びます。ただし，実際のところ，見た目では「1 : 1.6 = 5 : 8 の比を持つ長方形」と区別することはできないでしょう。「黄金比」が「美しく見える」として，「1 : 1.62 の方が 1 : 1.60 より美しく見える」と主張することには無理がありそうです。

縦 100 px, 横 162 px の長方形

縦 100 px, 横 160 px の長方形

フィボナッチ Fibonacci 数列は，「新しい項」が「前の項」と「さらにもう一つ前の項」の和で表される数列で,

$$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, \ldots$$

という並び方をします。となりあうフィボナッチ数の比は黄金比に近づくことが知られています。たとえば,

$$\frac{0}{1} = 0, \frac{1}{1} = 1, \frac{1}{2} = 0.5, \frac{2}{3} = 0.6666\cdots, \frac{3}{5} = 0.6, \frac{5}{8} = 0.625, \frac{8}{13} = 0.61538\cdots,$$
$$\frac{13}{21} = 0.61904\cdots,\ldots$$

などとなります。
第 n 番目のフィボナッチ数を F_n として，隣り合う２つの数の極限値を

$$x = \lim_{n\to\infty} \frac{F_{n-1}}{F_n} \tag{8.2.3}$$

とすれば,

$$x = \lim_{n\to\infty} \frac{F_{n-1}}{F_{n-1}+F_{n-2}} = \lim_{n\to\infty} \frac{1}{1+\frac{F_{n-2}}{F_{n-1}}} = \frac{1}{1+x} \tag{8.2.4}$$

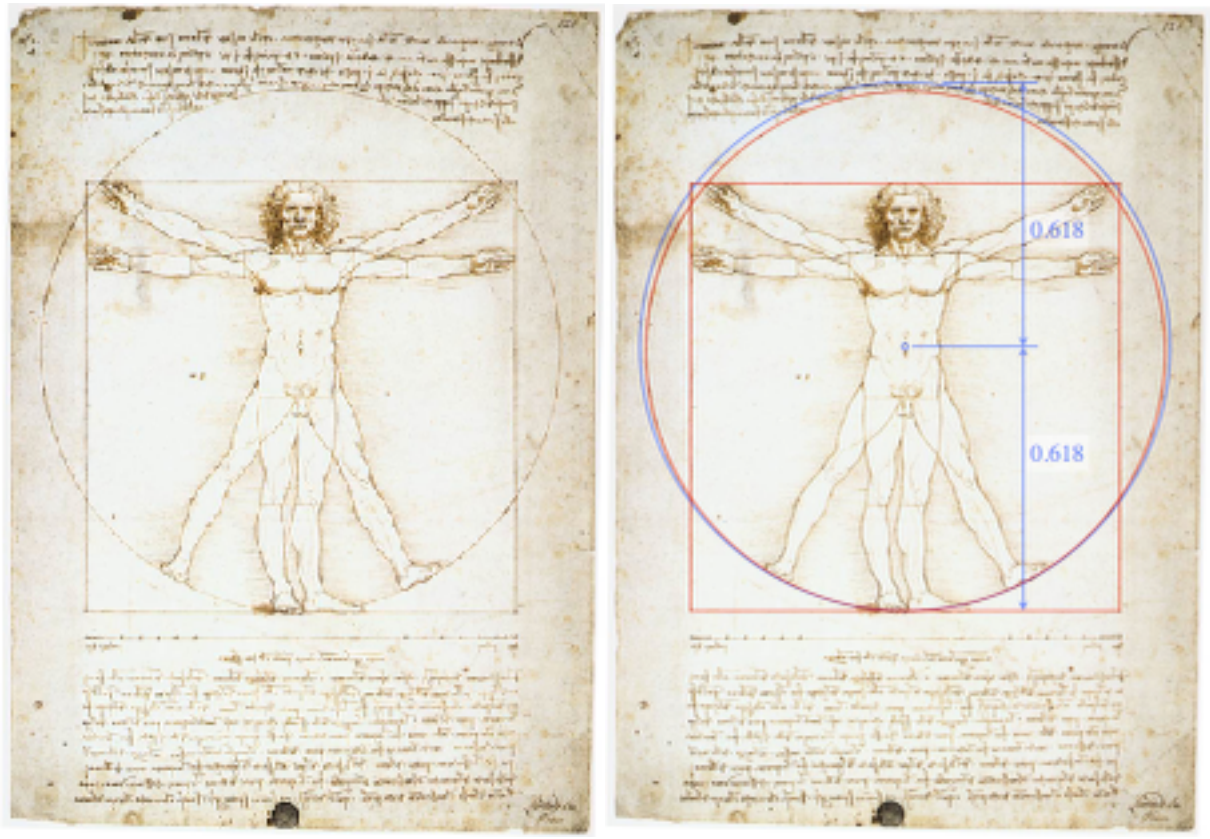
の関係から

$$x^2 + x - 1 = 0 \tag{8.2.5}$$

なので，$0 < x$ のとき

$$x = \frac{-1+\sqrt{5}}{2} = 0.618034\cdots \tag{8.2.6}$$

という解を持つことが自然に導かれます。



「自然や人工の造形，芸術作品に黄金比が隠されていることが多い」という人もいますが，「それほどでもない」という人も少なくありません。例えば，「レオナルド・ダヴィンチの『ウィトルウィウス的人体図』に描かれている円の半径と正方形の辺の長さの比は黄金比に等しい」という説がありますが，実際に黄金比に対応する円を描くと，右図の青線で示しているように，ダ・ヴィンチの描いた円（赤線）から，かなりずれていることがわかります。

【黄金分割アルゴリズム】

区間を $1-r : r$ あるいは $r : 1-r$ に分割することを黄金分割と呼びます。

黄金分割法は 1 次元の最小化アルゴリズムであり，比較的安定でプログラミングも容易です。以下のような手続きで，有限の区間内での関数 の最小を求めることができます。

(i) 区間の下限を a, 上限を d とする。$f(a)$,$f(d)$の値を計算する。

(ii) 以下の式に従って，区間を黄金分割する二つの点 b，c をとる。

$$b = ra + (1-r)d \tag{8.2.7}$$

$$c = (1-r)a + rd \tag{8.2.8}$$

また，$f(b)$，$f(c)$の値を計算する。

ここで，$\dfrac{b-a}{d-b} = \dfrac{1-r}{r}$，$\dfrac{c-a}{d-c} = \dfrac{r}{1-r}$ であり，b と c はそれぞれ区間 (a,d) を $1-r:r$, $r:1-r$ の比に内分する点である。

(iii) もし $f(b) < f(c)$ なら，最小となるのは (a,c) の区間内だから，以下の代入： $d \leftarrow c$, $c \leftarrow b$, $f(c) \leftarrow f(b)$ を行い，$b = ra + (1-r)d$ と $f(b)$ の値を計算する。$f(c) \le f(b)$ なら，最小となるのは (b,d) の区間内なので，以下の代入： $a \leftarrow b$，$b \leftarrow c$, $f(b) \leftarrow f(c)$ を行い，$c = (1-r)a + rd$ と $f(c)$ の値を計算する。

(iv) (iii) に戻る。

ただし，黄金分割法では常に正しい最小が導かれるとは限らず，局所的な極小 local minimum （偽の最小 false minimum）に陥る場合がありうることには注意するべきです。

例えば,

$$f(x) = -\exp\left[-0.2\left(x-0.5\right)^4\right] - \exp\left[-0.2\left(x-1.5\right)^2\right] \tag{8.2.9}$$

について，$x \in [0,2]$ として探索すると正しい最小位置 $x = 1.08843$ が求められますが,

$$f(x) = -\exp\left[-10\left(x-0.5\right)^4\right] - 1.2\exp\left[-10\left(x-1.5\right)^2\right] \tag{8.2.10}$$

については，$x \in [0,2]$ として黄金分割探索すると $x = 0.538108$ という偽最小に陥ってしまいます。探索範囲が $x \in [1,2]$ のように最小位置に近ければ，効率良く正しい最小位置が導かれます。

8－3　ネルダー・ミード法（滑降シンプレックス法）
Nelder-Mead method (Downhill simplex method)

ポリトープ polytope 法とも呼ばれます。二つ以上の変数を持つ関数の最適化で用いられます。コーディングに少し手間がかかりますが，動作は比較的安定・確実であり，効率も悪くなく，知っておくと良い方法です。

黄金分割法が一次元で「解を含む範囲を挟みうちしながら縮小していく」のと同じように，滑降シンプレックス法では「解を含む空間を囲む多面体を縮小していく」ような操作を繰り返します。

一般的に m 個のパラメータ（変数）を含む関数（m 次元関数）を最適化するためには，m 次元の空間（超空間）で $(m+1)$ 個の頂点を持つ多角形あるいは多面体（超多面体）を変形，移動する操作を行います。２変数の関数であれば，平面の上で三角形を使います。３変数の関数なら三次元空間で四面体を使います。４変数以上の関数であれば超空間で超多面体を使うことになります。

（１）各頂点 vertex の座標を $\mathbf{p}_i \ (i = 1, \cdots, m+1)$ とします。はじめに頂点の「初期配置」を決めます。事実上この初期配置の決め方で最適化がうまくいくか決まります。初期配置はユーザに指定させるのが普通ですが，m $(m+1)$ 個の数値を指定させることになり，このことでネルダー・ミード法が使いづらいものになる傾向があります。乱数を使ってランダムな配置から始める場合もあります。ユーザーに「粗い推定値」$\tilde{\mathbf{p}} = \left(\tilde{p}_1, \cdots, \tilde{p}_m\right)$ と「粗い推定誤差」$\Delta\tilde{\mathbf{p}} = \left(\Delta\tilde{p}_1, \cdots, \Delta\tilde{p}_m\right)$ の $2m$ 個の数値のみ指定させ，$(m+1)$ 個の頂点を $\tilde{\mathbf{p}}_1 = \left(\tilde{p}_1 + \Delta\tilde{p}_1, \cdots, \tilde{p}_m\right)$，$\tilde{\mathbf{p}}_m = \left(\tilde{p}_1, \cdots, \tilde{p}_m + \Delta\tilde{p}_m\right)$，$\tilde{\mathbf{p}}_{m+1} = \left(\tilde{p}_1, \cdots, \tilde{p}_m\right)$ のように配置するのも現実的な方法です。

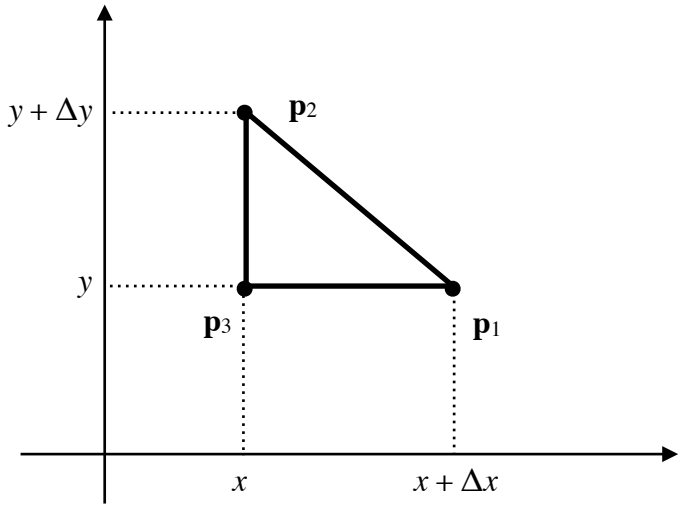


Fig. 8.3.1　初期シンプレックスの例

（２）各ステップ毎に，頂点を関数値に応じて序列化します。特に「最悪点」と「第二最悪点」「最良点」の３点の序列を決める必要があります。ここでは，

$$f\left(\mathbf{p}_1\right) \le f\left(\mathbf{p}_2\right) \le \cdots \le f\left(\mathbf{p}_m\right) \le f\left(\mathbf{p}_{m+1}\right)$$

の関係が成り立つとします。最良点が $\mathbf{p}_1$，第二最悪点が $\mathbf{p}_m$，最悪点が $\mathbf{p}_{m+1}$ であるとします。

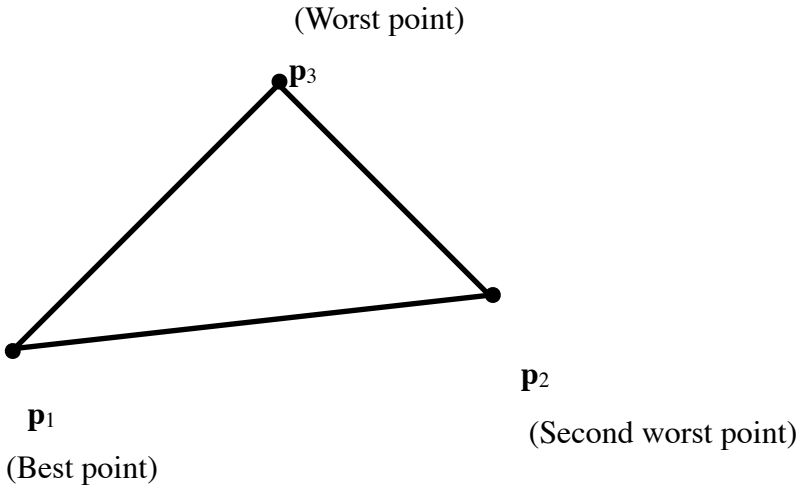


Fig. 8.3.2　頂点の序列化

以下の手順を繰り返します。

（３）最悪点 $\mathbf{p}_{m+1}$ 以外のすべての点の重心 gravity center：

$$\mathbf{p}_0 = \frac{1}{m}\sum_{i=1}^{m}\mathbf{p}_i$$

を求めます。

（４）最悪点 $\mathbf{p}_{m+1}$ を，重心 $\mathbf{p}_0$ に対して反射 reflection （幾何学の用語では反転 inversion に対応する）位置（反射点） $\mathbf{p}_{\mathrm{refl}}$ を求めます 。反射点は以下の式で得られます。

$$\mathbf{p}_{\mathrm{refl}} = \mathbf{p}_0 + \left(\mathbf{p}_0 - \mathbf{p}_{m+1}\right) = 2\mathbf{p}_0 - \mathbf{p}_{m+1}$$

また反射点での関数の値 $f\left(\mathbf{p}_{\mathrm{refl}}\right)$ を求めます。

（５）反射点 $\mathbf{p}_{\mathrm{refl}}$ が今までの最良点 $\mathbf{p}_1$ よりは悪いが，第二最悪点 $\mathbf{p}_m$ より良い場合，つまり

$$f\left(\mathbf{p}_1\right) \le f\left(\mathbf{p}_{\mathrm{refl}}\right) \le f\left(\mathbf{p}_m\right)$$

のとき，今までの最悪点 $\mathbf{p}_{m+1}$ を捨て，反射点 $\mathbf{p}_{\mathrm{refl}}$ に入れ替えます。このとき今までの第二最悪点が新しい最悪点になり最良点は変化しません。$\left\{\mathbf{p}_2, \cdots, \mathbf{p}_{m-1}\right\}$ と $\mathbf{p}_{\mathrm{refl}}$ とで新しく序列化を施します。ここでこのステップは終了し（３）に戻ります。

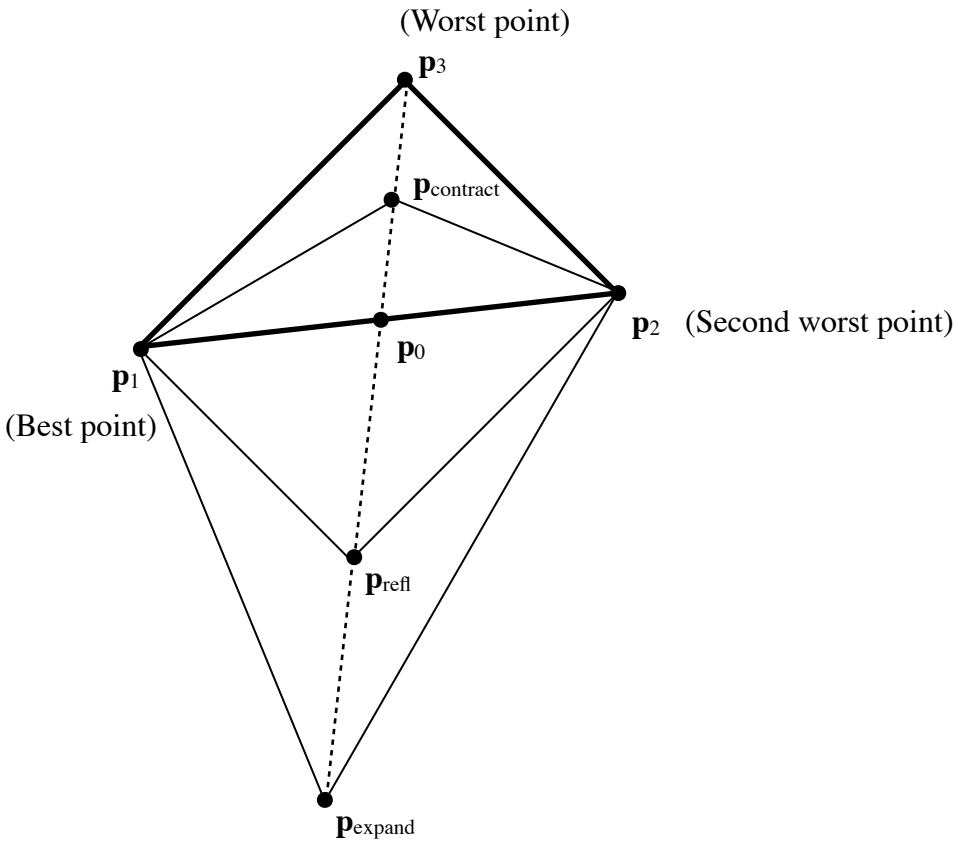


Fig. 8.3.3　最悪点 $\mathbf{p}_3$，最悪点以外の重心 $\mathbf{p}_0$，反射点 $\mathbf{p}_{\text{refl}}$，反射拡大点 $\mathbf{p}_{\text{expand}}$，収縮点 $\mathbf{p}_{\text{contract}}$ の関係

（６）反射点 $\mathbf{p}_{\text{refl}}$ が今までの最良点 $\mathbf{p}_1$ より良い場合，つまり，$f\left(\mathbf{p}_{\text{refl}}\right) < f\left(\mathbf{p}_1\right)$

のとき，反射拡大 reflection & expansion 点 $\mathbf{p}_{\text{expand}}$を求め，関数値$f\left(\mathbf{p}_{\text{expand}}\right)$ を計算します。反射拡大点は次式で得られます。

$$\mathbf{p}_{\text{expand}} = \mathbf{p}_0 + 2\left(\mathbf{p}_0 - \mathbf{p}_{m+1}\right) = 3\mathbf{p}_0 - 2\mathbf{p}_{m+1}$$

（７）反射拡大点 $\mathbf{p}_{\text{expand}}$ が反射点 $\mathbf{p}_{\text{refl}}$ より良い場合，つまり

$$f\left(\mathbf{p}_{\text{expand}}\right) < f\left(\mathbf{p}_{\text{refl}}\right) < f\left(\mathbf{p}_1\right)$$

のとき，最悪点 $\mathbf{p}_{m+1}$ を捨て，反射拡大点 $\mathbf{p}_{\text{expand}}$ を採用し，序列を更新します。ここでこのステップは終了し，（３）に戻ります。

（８）反射拡大点 $\mathbf{p}_{\text{expand}}$ が反射点 $\mathbf{p}_{\text{refl}}$ より悪い場合，つまり

$$f\left(\mathbf{p}_{\text{refl}}\right) \le f\left(\mathbf{p}_{\text{expand}}\right)$$

のとき，最悪点 $\mathbf{p}_{m+1}$ を捨て，反射点 $\mathbf{p}_{\text{refl}}$ を採用し，序列を更新します。ここでこのステップは終了し，（３）に戻ります。

（９）反射点が第二最悪点より悪い場合，つまり

$$f\left(\mathbf{p}_{m}\right) \le f\left(\mathbf{p}_{\text{refl}}\right)$$

のとき，一次元収縮 contraction を施し収縮点 $\mathbf{p}_{\text{contract}}$ を求め，関数値 $f\left(\mathbf{p}_{\text{contract}}\right)$ を計算します。収縮点は以下の式で得られます。

$$\mathbf{p}_{\text{contract}} = \mathbf{p}_0 - 0.5\left(\mathbf{p}_0 - \mathbf{p}_{m+1}\right) = 0.5\mathbf{p}_0 + 0.5\mathbf{p}_{m+1}$$

（１０）収縮点 $\mathbf{p}_{\text{contract}}$ が最悪点 $\mathbf{p}_{m+1}$ より良い場合，つまり

$$f\left(\mathbf{p}_{\text{contract}}\right) \le f\left(\mathbf{p}_{m+1}\right)$$

のとき，最悪点 $\mathbf{p}_{m+1}$ を捨て，収縮点 $\mathbf{p}_{\text{contract}}$ を採用し，頂点の序列を更新します。ここでこのステップは終了し（３）に戻ります。

（１１）収縮点 $\mathbf{p}_{\text{contract}}$ が最悪点 $\mathbf{p}_{m+1}$ より悪い場合，つまり

$$f\left(\mathbf{p}_{m+1}\right) < f\left(\mathbf{p}_{\text{contract}}\right)$$

のとき，最良点 $\mathbf{p}_1$ を中心に縮小 reduction を施します。「縮小」の操作は「反射」「反射拡大」「収縮」とは異なり，最良点以外の全ての点に施される操作で，以下の式で表されます。

$$\mathbf{p}_i \leftarrow \mathbf{p}_i - 0.5\left(\mathbf{p}_i - \mathbf{p}_1\right) = 1.5\ \mathbf{p}_i - 0.5\ \mathbf{p}_1$$

$$(i = 2,3,\cdots,m+1)$$

関数値 。

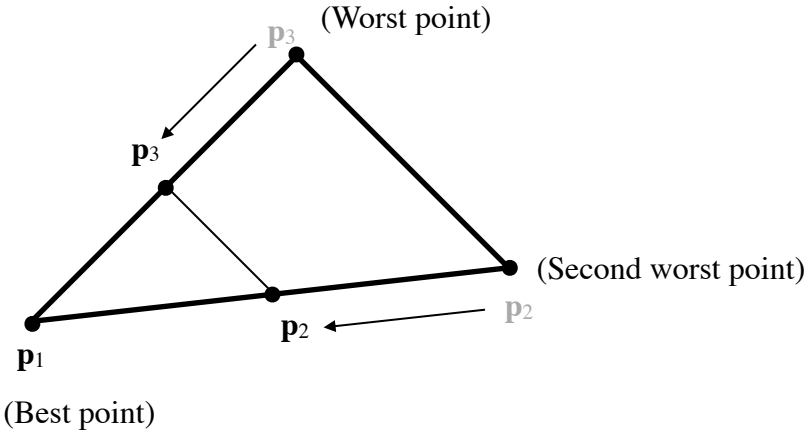


Fig. 8.3.4　ネルダー・ミード法の「縮小 reduction」操作

ネルダー・ミード法で「極小」が求まることは理論的に証明されていますが，シンプレックスの初期配置によって「偽最小」に陥る場合があることは，黄金分割法と同じです。